Review
- Single Pixel Filters
  - Thresholding
  - Posterize
  - Histogram Equalization
  - Negative
  - Sepia
  - Grayscale
- Spatial Filters
  - Smooth
  - Blur — Low Pass Filter
  - Sharpen — High Pass Filter
  - Erosion
  - Dilation
- Image Processing Applications

What's a string?
Characters enclosed by double quotes
"this is a String"
" this String starts with spaces"
"12345"
"the above String is made up of digit characters"

Print Strings to the Console using println()
println( "The mouse was pressed" );

Strings are Objects
Defined using a class
Have fields, methods, one or more constructors

String objects hold an array of 'chars'

What's a char?
A character enclosed by single quotes ('A')

```java
String msg = "I Love CS 110!";
```

<table>
<thead>
<tr>
<th>Type</th>
<th>Range</th>
<th>Default</th>
<th>Bytes</th>
</tr>
</thead>
<tbody>
<tr>
<td>boolean</td>
<td>{ true, false }</td>
<td>false</td>
<td>7</td>
</tr>
<tr>
<td>byte</td>
<td>{ 0..255 }</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>int</td>
<td>{ -2,147,483,648 .. 2,147,483,647 }</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>long</td>
<td>{ -9,223,372,036,854,775,808 .. 9,223,372,036,854,775,807 }</td>
<td>0</td>
<td>8</td>
</tr>
<tr>
<td>float</td>
<td>{ -3.40282347E+38 .. 3.40282347E+38 }</td>
<td>0.0</td>
<td>4</td>
</tr>
<tr>
<td>double</td>
<td>much larger/smaller</td>
<td>0.0</td>
<td>8</td>
</tr>
<tr>
<td>color</td>
<td>{ #00000000 .. #FFFFFFFF }</td>
<td>black</td>
<td>4</td>
</tr>
<tr>
<td>char</td>
<td>a single character 'a', 'b', ... 'u0000'</td>
<td></td>
<td>2</td>
</tr>
</tbody>
</table>

Making Strings
- Declaring String objects with no chars
  ```java
  String myName;
  String myName = new String();
  ```

- Declaring String objects init'd w/ char array
  ```java
  String myName = "Fred";
  String myName = new String("Fred");
  ```

Chars are encoded by bytes

ASCII
- American Standard Code for Information Interchange
- An early character encoding standard
- glyph <-> byte mapping
- 127 characters
- Forms the basis of new encoding standards
- Unicode: more than 109,000 characters covering 93 scripts

Note:
- Numbers are different than the digit characters
- Includes special characters and punctuation
String class methods

- `charAt(index)`
  - Returns the character at the specified index
- `equals(anotherString)`
  - Compares a string to a specified object
  - `EqualsIgnoreCase(anotherString)`
    - SA ignoring case (i.e. 'A' == 'a')
- `indexOf(char)`
  - Returns the index value of the first occurrence of a character within the input string
- `length()`
  - Returns the number of characters in the input string
- `substring(startIndex, endIndex)`
  - Returns a new string that is part of the input string
- `toLowerCase()`
  - Converts all the characters to lower case
- `toUpperCase()`
  - Converts all the characters to upper case
- `concat(anotherString)`
  - Concatenates String with anotherString

Comparing Strings: Always use equals()

- Never use `==` ... Why?
  - String are objects
  - The `==` operator checks that two items are identical
  - Two objects can contain the same data, but be different object instances
  - The `==` operator tests that the two objects are the same object ... generally, that's not what we want
  - The equals() method tests the data of the two String objects for equality

Other forms of `indexOf()`

<table>
<thead>
<tr>
<th>Returns</th>
<th>Description</th>
</tr>
</thead>
</table>
| int     | `indexOf(int ch)`
  - Returns the index within this string of the first occurrence of the specified character. |
| int     | `indexOf(int ch, int fromIndex)`
  - Returns the index within this string of the first occurrence of the specified character, starting the search at the specified index. |
| int     | `indexOf(String str)`
  - Returns the index within this string of the first occurrence of the specified substring. |
| int     | `indexOf(String str, int fromIndex)`
  - Returns the index within this string of the first occurrence of the specified substring, starting at the specified index. |

Other forms of `substring()`

<table>
<thead>
<tr>
<th>Returns</th>
<th>Description</th>
</tr>
</thead>
</table>
| String  | `substring(int beginIndex)`
  - Returns a new string that is a substring of this string. |
| String  | `substring(int beginIndex, int endIndex)`
  - Returns a new string that is a substring of this string.
Digit characters in a String are not integers

```java
String s = "12345";
void setup() {
    char myChar = s.charAt(1);
    byte myByte = byte(myChar);
    println(myByte);
}
```

Building Strings – Use '+'

```java
void setup() {
    String s1 = "Hello";
    String s2 = "World";
    String s3 = one + " " + two;
    println( s3 );
}
```

```java
void setup() {
    String s1 = "She is number ";
    String s2 = " in computer science.";
    String s3 = s1 + 1 + s2;
    println( s3 );
}
```

```java
void setup() {
    println("This is line 1
This is line 2");
}
```

```java
void setup() {
    println("This is line 1\nThis is line 2");
}
```

Strings can be held by Arrays

```java
String[] tokens = new String[5];
void setup() {
    tokens[0] = "one";
    tokens[1] = "two";
    tokens[2] = "three";
    tokens[3] = "four";
    tokens[4] = "five";
    println(tokens);
}
```

```java
String[] tokens = new String[5];
void setup() {
    println(tokens);
}
```

```java
String[] tokens = new String[5];
void setup() {
    println(tokens);
}
```

```java
String[] tokens = new String[5];
void setup() {
    println(tokens);
}
```

```java
String[] tokens = new String[5];
void setup() {
    println(tokens);
}
```
### Built-in String functions (not methods)

- **split(String bigString, String splitChar)**
  - Breaks a String into an array, splitting on splitChar.
  - Returns new String array.

- **splitTokens(String bigString, String splitCharString)**
  - Breaks a string into an array, splitting on any char in `splitCharString`.
  - Returns new String array.

- **join(String[] stringArray, String joinChar)**
  - Builds a new String by concatenating all Strings in `stringArray`, placing `joinChar` between each.
  - Inverse of `split()` function.

- **nf(Integer intValue, Integer digits)**
  - Formats a number as a String.

- **trim(String theString)**
  - Removes whitespace from the beginning and end of `theString`.

- **text(String theString, int x, int y)**
  - Draws `theString` on the sketch at `(x, y)`.

### Split a String based on a single or multiple separator chars

```java
String s1 = "Data: 12, 34, 56";
String[] as;
void setup() {
  as = split(s1, ",");
  //as = trim(as);
  println(as);
}
```

|0| "12"
|1| "34"
|2| "56"

```java
String s1 = "12, 34, 56";
String[] as;
void setup() {
  as = splitTokens(s1, ",");
  //as = trim(as);
  println(as);
}
```

|0| "Data"
|1| "12"
|2| "34"
|3| "56"

### Join a String Array with a join char

```java
String[] as = new String[] {"one", "two", "buckle my shoe"};
void setup() {
  String s1 = join(as, " | ");
  println(s1);
}
```

```
one | two | buckle my shoe```

### Numbers can be formatted as Strings

```java
phrase = s1 + nf(7, 3) + " " + s2;
// nf(integer, number of digits)
// "She is the 007 programmer."
```

```java
phrase = s1 + nf(3.14159, 3, 2) + " " + s2;
// nf(float, digits before decimal, digits after decimal)
// "She is the 03.14 programmer."
```

```
String stripSpaces(String s) {
  for (int i=s.length()-1; i>=0; i--) {
    if (s.charAt(i) == ' ') {
      s = s.substring(0, i) + s.substring(i+1);
    }
  }
  return s;
}
```

### Remove spaces from a String